

DEVELOPMENT OF A SELF-BALANCED ROBOT & ITS CONTROLLER

A REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF TECHNOLOGY

IN

MECHANICAL ENGINEERING

BY

SUSHIL GARG



**DEPARTMENT OF MECHANICAL ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY
ROURKELA-769008**

2010

DEVELOPMENT OF A SELF-BALANCED ROBOT & ITS CONTROLLER

A REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF TECHNOLOGY

IN

MECHANICAL ENGINEERING

BY

SUSHIL GARG

Under the guidance of

Prof. DAYAL R. PARHI



**DEPARTMENT OF MECHANICAL ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY
ROURKELA-769008**

2010



**NATIONAL INSTITUTE OF TECHNOLOGY
ROURKELA**

CERTIFICATE

This is to certify that the Project report entitled “Development of self-balanced robot & its controller” submitted by Sri Sushil Garg in partial fulfillment of the requirements for the award of Bachelor of technology Degree in Mechanical Engineering at the National Institute of Technology, Rourkela (Deemed University) is an authentic work carried out by him under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University / Institute for the award of any Degree or Diploma.

DATE: 13/05/2010
PLACE: ROURKELA

Prof. DAYAL R. PARHI
Mechanical Engineering Department
National Institute of Technology
Rourkela-769008

ACKNOWLEDGEMENT

I would like to express my deep sense of gratitude and respect to my supervisor Prof. Dayal R. Parhi for his excellent guidance, suggestions and support. I consider myself extremely lucky to be able to work under the guidance of such a dynamic personality.

I would like to render heartiest thanks to my friend who's ever helping nature and suggestion has helped us to complete this present work.

DATE: 13-05-2010
PLACE: ROURKELA

SUSHIL GARG
Roll No. - 10603036
8th Semester, B. Tech.
Mechanical Engineering Department
National Institute of Technology
Rourkela-769008

ABSTRACT

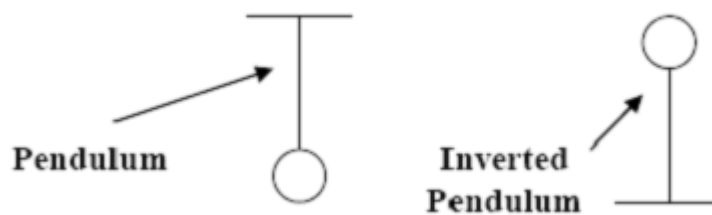
Two wheeled balancing robots are based on inverted pendulum configuration which relies upon dynamic balancing systems for balancing and maneuvering. This project is based on the development of a self-balanced two wheeled robot which has a configuration similar to a bicycle. These robot bases provide exceptional robustness and capability due to their smaller size and power requirements. Outcome of research in this field had led to the birth of robots such as Segway, Murata boy etc. Such robots find their applications in surveillance & transportation purpose. Here, in particular, the focus is on the electro-mechanical mechanisms & control algorithms required to enable the robot to perceive and act in real time for a dynamically changing world. Using an Ultrasonic sensor and an accelerometer we get the information about the tilt of the robot from its equilibrium position. Balancing was done using a servo motor, a DC motor and a control momnt gyroscope. While these techniques are applicable to many robot applications, the construction of sensors, filters and actuator system is a learning experience.

CONTENT

S. No.	TOPIC	Page No.
01	INTRODUCTION	01
02	LITERATURE REVIEW	03
03	INERTIAL SENSOR UNIT	08
04	LOGICAL PROCESSING UNIT	12
05	ACTUATOR UNIT	21
06	CONCLUSION	24
07	REFERENCES	25
08	APPENDICES	29

INTRODUCTION

Balancing of a two-wheeled Robot is a classic engineering problem. It is based on inverted pendulum problem and is much like trying to balance a broom or a rod on the tip of your finger. This challenging electronics, robotics and controls problem is the basis of my study for this project.



Balancing Process

The word balance means that the inverted pendulum is in equilibrium state, in which its position is like standing upright 90 degrees. However, the system itself is not balance, which means it keeps falling off, away from the vertical axis. Using an ultrasonic sensor, this deviation from the equilibrium position is measured and fed into the microcontroller, where the program in itself is a balancing algorithm. The microcontroller will then provide a type of feedback signal through PWM control to the H-bridge circuit to turn the servo motor attached to the control moment gyroscope clockwise or anticlockwise, thus balancing the robot.

The code is written in C language and compiled for the Atmel ATmega16 or AVR microcontroller, which is interfaced with the sensors (ultrasonic and accelerometer) and motors. The main goal of the microcontroller is to fuse the wheel encoder, ultrasonic and accelerometer sensors to estimate the attitude of the platform and then to use this information to drive the control moment gyroscope in the direction to maintain an upright and balanced position. The basic concept for a two-wheeled dynamically balancing robot is pretty simple: move the actuator in a direction to counter the direction of fall. In practice this requires two feedback sensors: a position sensor to measure the tilt of the robot, an accelerometer to calibrate the ultrasonic. These two measurements are summed and fed back to the actuator which produces the counter torque required to balance the robot.

The robot can be classified into the following parts:

- Inertial sensors
- Logical processing unit
- Actuator

LITERATURE REVIEW

This section provides an insight and literature review to the current technology available to construct a two-wheel self balancing robot. It also highlights various methods used by researches on this topic.

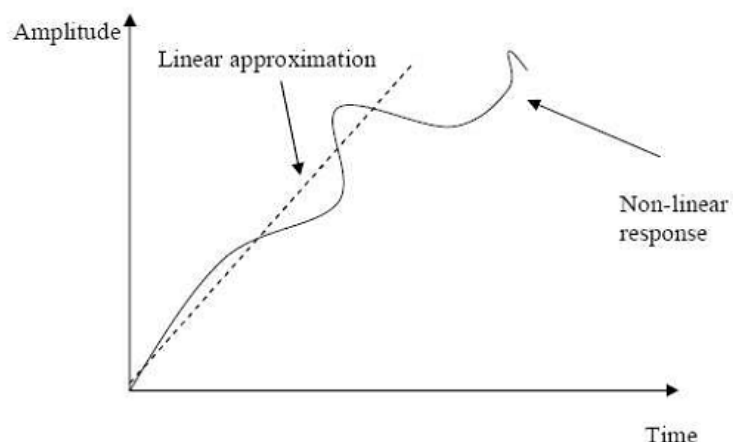
Balancing robots

The concept of robot balancing is based on the inverted pendulum model. This model has been widely used by designers and researchers around the world in controlling a system not only in designing wheeled robot but also other types of robot as well such as legged robots. Researchers at the Industrial Electronics Laboratory at the Swiss Federal Institute of Technology, Switzerland have built a prototype two wheel robot in which the control is based on a Digital Signal Processor. A linear state space controller using information from a gyroscope and motor encoder sensors is being implemented to make this system stabilize. (Grasser et al.2002). Another two wheeled robot called ‘SEGWAY HT’ is available commercially (Dean Kamen, 2001). It is invented by Dean Kamen who has design more than 150 systems including climate control systems and helicopter design. An extra feature this robot has is that it is able to balance itself while a user is standing on top of it and navigate the terrain with it. This robot uses five gyroscopes and a few other tilt sensors to keep it balanced. Next is the small scale robot, Nbot which is similar to JOE is built by David. P Anderson. (Anderson, David.P) This robot uses a commercially available inertial sensor and position information from motor encoder for balancing of the system. This robot has won the NASA cool robot of the week in the year 2003.

Steven Hassenplug used a more innovative approach to construct a balancing robot (Steve Hassenplug, 2002). The chassis of the body is constructed by using the LEGO Mindstorms robotics kit. The balancing method of controlling the system is unique with two Electro-Optical Proximity Detector sensors is used to provide the tilt angle information for the controller. This omits the conventional use of gyroscope that has been used by previous robot researchers. Louis Brennan, an Irish-Australian inventor, was one of the first to patent a gyroscopic stabilizing vehicle. In 1903, Brennan patented a gyroscopically balanced monorail system that he designed for military use; he successfully demonstrated the apparatus in 1909. By mounting one or more gyrostats (a modified gyroscope) along the body, the monorail balanced itself when its equilibrium was disturbed. Brennan feared that the gyrostats would fail in use, causing total system failure; thus, he prevented the monorail from being mass-produced. More recently, a group from Columbia University manufactured a modernized version of Brennan's monorail. Unfortunately, the group was unable to create a working model. The electronic component of the model continuously overheated during operation, causing the motor to burn out. The electronic segment was improperly modeled, which led to the mechanism's inability to perform.

Control System

Over the years, there are only two types of control being used by researchers in controlling a system. The types of control can be categorized as linear and non-



linear control. In some instances, the linear control is sufficient for controlling a system. One of the most widely used is the Proportional Derivative Integral controller or better known as the PID controller (Rick Bickle, 2003). The others are linear quadratic controller (LQR), fuzzy logic controller, pole placement controller etc. It is generally accepted that linear control is more popular than non-linear control. There are two reasons for this. In all cases, the modeling of a system requires a lot of parameters to be considered and applied and this makes the system complex. However, some of the parameters values needed to model the system are small. Hence, most researchers would prefer to model their applications in a linear approximation, which is simpler and in some instances effective. However, in most cases the linear control theory is not suitable for real life implementation, which mostly exhibit non-linear response. For better performance some non-linear approximation can be applied. (J.J. D'Azzo, pg 11). Figure here shows how a non-linear response can be approximated to a linear response.

Data Acquisition

In a paper 'Attitude Estimation Using Low Cost Accelerometer and gyroscope' presented by Young Soo Suh, he shows the two different sensors which are the accelerometer and gyroscope that exhibits poor results when use separately to determine the attitude which is referred as the pitch angle or roll angle. The factor that contributes to the deviation of the desired result of the gyroscope is due to the drift term. Since the drift increases with time, error in output data will also increase. One of the disadvantages of using accelerometer individually is that the device is sensitive to vibration as vibration contains lot of acceleration components. One solution that Young suggested is that a low pass filter is required to limit the high frequency. However,

the gyroscope can combine with accelerometer to determine the pitch or roll angle with much better result with the use of Kalman filter.

Kalman Filter

The purpose of this filter is to solve problems of statistical nature (Kalman, 1960). It is a mathematical method named after Rudolf E. Kalman. Its purpose is to use measurements that are observed over time that contain noise (random variations) and other inaccuracies, and produce values that tend to be closer to the true values of the measurements and their associated calculated values. The Kalman filter has many applications in technology, and is an essential part of the development of space and military technology. Perhaps the most commonly used type of very simple Kalman filter is the phase-locked loop, which is now ubiquitous in FM radios and most electronic communications equipment. Extensions and generalizations to the method have also been developed.

The Kalman filter produces estimates of the true values of measurements and their associated calculated values by predicting a value, estimating the uncertainty of the predicted value, and computing a weighted average of the predicted value and the measured value. The most weight is given to the value with the least uncertainty. The estimates produced by the method tend to be closer to the true values than the original measurements because the weighted average has a better estimated uncertainty than either of the values that went into the weighted average.

The Kalman filter uses a system's dynamics model (i.e., physical laws of motion), known control inputs to that system, and measurements (such as from sensors) to form an estimate of the

system's varying quantities (its state) that is better than the estimate obtained by using any one measurement alone. As such, it is a common sensor fusion algorithm.

All measurements and calculations based on models are estimates to some degree. Noisy sensor data, approximations in the equations that describe how a system changes, and external factors that are not accounted for introduce some uncertainty about the inferred values for a system's state. The Kalman filter averages a prediction of a system's state with a new measurement using a weighted average. The purpose of the weights is that values with better estimated uncertainty are "trusted" more. The weights are calculated from the covariance, a measure of the estimated uncertainty of the prediction of the system's state. The result of the weighted average is a new state estimate that lies in between the predicted and measured state, and has a better estimated uncertainty than either alone. This process is repeated every time step, with the new estimate and its covariance informing the prediction used in the following iteration. This means that the Kalman filter works recursively and requires only the last "best guess" - not the entire history - of a system's state to calculate a new state.

When performing the actual calculations for the filter (as discussed below), the state estimate and covariances are coded into matrices to handle the multiple dimensions involved in a single set of calculations. This allows for representation of linear relationships between different state variables (such as position, velocity, and acceleration) in any of the transition models or covariances.

The Kalman filter is used in sensor fusion and data fusion. Typically real time systems produce multiple sequential measurements rather than making a single measurement to obtain the state of the system. These multiple measurements are then combined mathematically to generate the system's state at that time instance.

INERTIAL SENSOR UNIT

The inertial sensors used here are:

- Ultrasonic Sonar sensor
- Freescale 1-axis accelerometer

Ultrasonic sensors (also known as transceivers when they both send and receive) work on a principle similar to radar or sonar which evaluate attributes of a target by interpreting the echoes from radio or sound waves respectively. Ultrasonic sensors generate high frequency sound waves and evaluate the echo which is received back by the sensor. Sensors calculate the time interval between sending the signal and receiving the echo to determine the distance to an object. Systems typically use a transducer which generates sound waves in the ultrasonic range, above 20,000 hertz, by turning electrical energy into sound, then upon receiving the echo turn the sound waves into electrical energy which can be measured and displayed.

An ultrasonic transducer is a device that converts energy into ultrasound, or sound waves above the normal range of human hearing. While technically a dog whistle is an ultrasonic transducer that converts mechanical energy in the form of air pressure into ultrasonic sound waves, the term is more apt to be used to refer to piezoelectric transducers that convert electrical energy into sound. Piezoelectric crystals have the property of changing size when a voltage is applied, thus

applying an alternating current (AC) across them causes them to oscillate at very high frequencies, thus producing very high frequency sound waves.

The location, at which a transducer focuses the sound, can be determined by the active transducer area and shape, the ultrasound frequency and the sound velocity of the propagation medium.

Since piezoelectric crystals generate a voltage when force is applied to them, the same crystal can be used as an ultrasonic detector. Some systems use separate transmitter and receiver components while others combine both in a single piezoelectric transceiver. Alternative methods for creating and detecting ultrasound include magnetostriction and capacitive actuation.

For the current work, One ultrasonic sensor is used which is mounted near the paddle of the bicycle and gives the distance from the ground as the output. As the cycle tilts in either direction, the distance from the ground changes and using ultrasonic sensor, we can detect the amount of tilt.

Software will combine this measurement with that of an accelerometer to deduce a better estimate of absolute position. The accelerometer does give a physical reference because it is able to measure the static gravitational force.

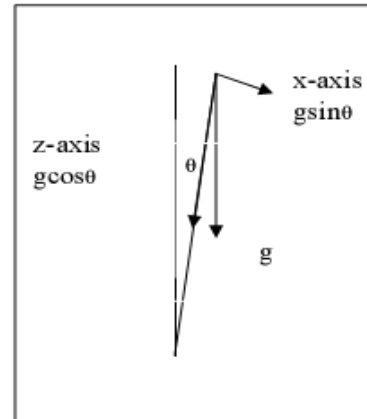
The accelerometer is free from drift and errors due to integration. The angle of tilt can be calculated by measuring the acceleration along x-direction.

$$A_x = g \sin \theta$$

For small values of θ ,

$$A_x = g \theta$$

$$\theta = K_i * (A_x) \quad ; K_i \text{ is a constant}$$



This approximation is only accurate for small values of θ ;

considering our tilt to be small, we go ahead with this approximation as inverse trigonometric functions will be time consuming for an 8-bit microcontroller.

This is the case considering the static acceleration. In case of dynamic acceleration, the equations are modified as;

$$A_x = g \sin \theta - \partial V_x / \partial t$$

Considering small angles;

$$\theta = K_i * (A_x + r\dot{\omega})$$

For calculating the integrals and derivatives, the PID algorithm is used.

Calculation of θ_t by integration;

$$\theta_t = \theta_{t-1} + \omega * \delta t$$

Calculation of $\dot{\omega}$ by differentiation;

$$\dot{\omega}_t = (\omega_t - \omega_{t-1}) / \delta t$$

Following code was used for interfacing of ultrasonic sensor and getting the output of the sensor onto the LCD:

```
#include<avr/io.h>
#include<avr/delay.h>
#include<avr/interrupt.h>
#include<compat/deprecated.h>
#include "C:\Documents and Settings\Administrator\My Documents\LCD\lcd.h"
#include "C:\Documents and Settings\Administrator\My Documents\LCD\lcd.c"

int main(void)
{
    DDRB=0xFB;

    DDRA=0xFF;

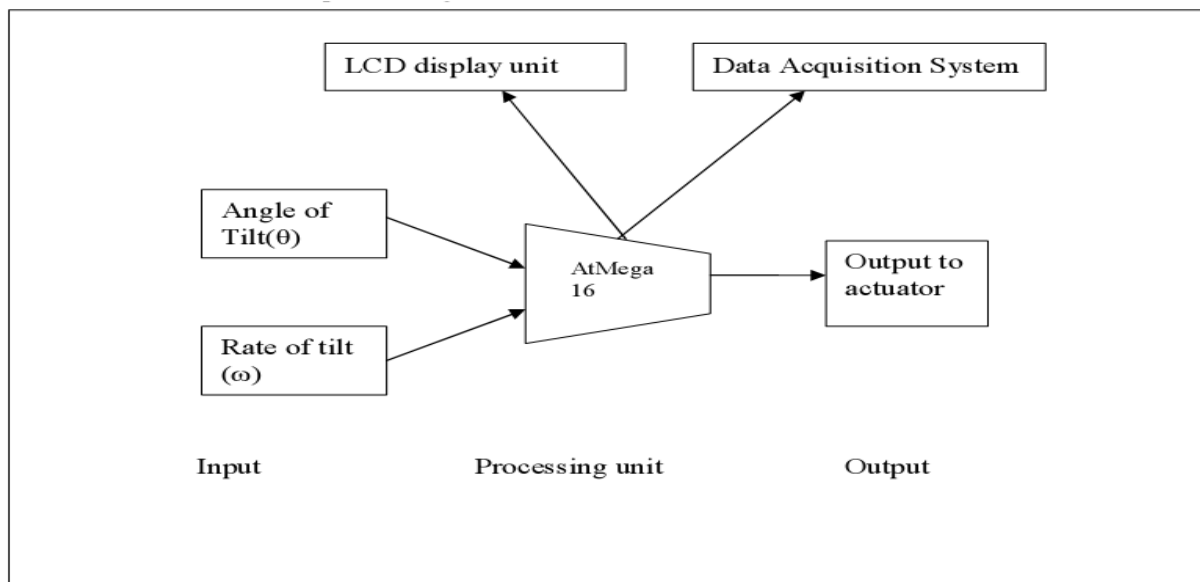
    DDRC=0xFF;
    int i;
    lcd_init(LCD_DISP_ON);
    void lcd_send(int x)
    {
        char buffer[10];
        itoa(x , buffer, 10);
        lcd_puts(buffer);
        lcd_puts(" ");
    }
    {
        TCCR1B |= (1 << CS10); // Set up timer
    }
    while(1)
    {
        DDRA=0xF8;
        i=0;
        sbi(PORTA,3);
        _delay_us(5);
        DDRA=0xF0;
        cbi(PORTA,3);
        while(bit_is_clear(PINA,3));
        TCNT1=0;
        while(bit_is_set(PINA,3));
        i=TCNT1;
        lcd_clrscr();
        lcd_gotoxy(0,1);
        lcd_send(i);
    }
}
```

LOGICAL PROCESSING UNIT

The processing unit used is Atmel ATmega16, 8-bit microcontroller unit which is a versatile EEPROM. It has four I/O ports, onboard ADC and two PWM outputs. It can be programmed easily with minimum hardware requirements which make it extremely popular in robotics applications. Here it performs the following functions:

- ADC conversion of outputs of Ultrasonic Sensor and Accelerometer
- Processing the input signals
- Display of Data
- Control of actuator unit

Schematic of the processing unit is as shown-



The microcontroller is clocked at 12MHz frequency by connecting a crystal across pins 12-13.
The complete specification is mentioned in appendix.

Details of the hardware:

Operating system: Microsoft XP service Pack 2

Programming platform: WinAVR, Atmel Studio

Programmer: AVRdude.

Microcontroller: ATmega16L, 40 PIN DIP

PID CONTROLLER

A proportional–integral–derivative controller (PID controller) is a generic control loop feedback mechanism (controller) widely used in industrial control systems. A PID controller attempts to correct the error between a measured process variable and a desired setpoint by calculating and then instigating a corrective action that can adjust the process accordingly and rapidly, to keep the error minimal.

The PID controller calculation (algorithm) involves three separate parameters; the proportional, the integral and derivative values. The proportional value determines the reaction to the current error, the integral value determines the reaction based on the sum of recent errors, and the derivative value determines the reaction based on the rate at which the error has been changing.

The weighted sum of these three actions is used to adjust the process via a control element such

as the position of a control valve or the power supply of a heating element. By tuning the three constants in the PID controller algorithm, the controller can provide control action designed for specific process requirements. The response of the controller can be described in terms of the responsiveness of the controller to an error, the degree to which the controller overshoots the setpoint and the degree of system oscillation. Note that the use of the PID algorithm for control does not guarantee optimal control of the system or system stability. Some applications may require using only one or two modes to provide the appropriate system control. This is achieved by setting the gain of undesired control outputs to zero. A PID controller will be called a PI, PD, P or I controller in the absence of the respective control actions. PI controllers are particularly common, since derivative action is very sensitive to measurement noise, and the absence of an integral value may prevent the system from reaching its target value due to the control action.

A familiar example of a control loop is the action taken to keep one's shower water at the ideal temperature, which typically involves the mixing of two process streams, cold and hot water. The person feels the water to estimate its temperature. Based on this measurement they perform a control action: use the cold water tap to adjust the process. The person would repeat this input-output control loop, adjusting the hot water flow until the process temperature stabilized at the desired value. Feeling the water temperature is taking a measurement of the process value or process variable (PV). The desired temperature is called the setpoint (SP). The output from the controller and input to the process (the tap position) is called the manipulated variable (MV). The difference between the measurement and the setpoint is the error (e), too hot or too cold and by how much. As a controller, one decides roughly how much to change the tap position (MV) after one determines the temperature (PV), and therefore the error. This first estimate is the equivalent of the proportional action of a PID controller. The integral action of a PID controller can be thought of as gradually adjusting the temperature when it is almost right. Derivative action can

be thought of as noticing the water temperature is getting hotter or colder, and how fast, anticipating further change and tempering adjustments for a soft landing at the desired temperature (SP).

The PID control scheme is named after its three correcting terms, whose sum constitutes the manipulated variable (MV). Hence:

$$MV(t) = P_{out} + I_{out} + D_{out}$$

Where,

P_{out} , I_{out} , and D_{out} are the contributions to the output from the PID controller from each of the three terms, as defined below. The proportional, integral, and derivative terms are summed to calculate the output of the PID controller. Defining $u(t)$ as the controller output, the final form of the PID algorithm is:

$$u(t) = MV(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{d}{dt} e(t)$$

Where the tuning parameters are:

Proportional gain, K_p

Larger values typically mean faster response since the larger the error, the larger the proportional term compensation. An excessively large proportional gain will lead to process instability and oscillation.

Integral gain, K_i

Larger values imply steady state errors are eliminated more quickly. The trade-off is larger overshoot: any negative error integrated during transient response must be integrated away by positive error before reaching steady state.

Derivative gain, K_d

Larger values decrease overshoot, but slow down transient response and may lead to instability due to signal noise amplification in the differentiation of the error.

Pseudo code

Here is a simple software loop that implements the PID algorithm:

```
previous_error = 0
integral = 0
start:
    error = setpoint - actual_position
    integral = integral + (error*dt)
    derivative = (error - previous_error)/dt
    output = (Kp*error) + (Ki*integral) + (Kd*derivative)
    previous_error = error
    wait(dt)
    goto start
```

FUZZY LOGIC

The concept of Fuzzy Logic (FL) was conceived by Lotfi Zadeh, a professor at the University of California at Berkley, and presented not as a control methodology, but as a way of processing data by allowing partial set membership rather than crisp set membership or non-membership. This approach to set theory was not applied to control systems until the 70's due to insufficient small-computer capability prior to that time. Professor Zadeh reasoned that people do not require precise, numerical information input, and yet they are capable of highly adaptive control. If feedback controllers could be programmed to accept noisy, imprecise input, they would be much more effective and perhaps easier to implement.

FL is a problem-solving control system methodology that lends itself to implementation in systems ranging from simple, small, embedded micro-controllers to large, networked, multi-channel PC or workstation-based data acquisition and control systems. It can be implemented in hardware, software, or a combination of both. FL provides a simple way to arrive at a definite conclusion based upon vague, ambiguous, imprecise, noisy, or missing input information. FL's approach to control problems mimics how a person would make decisions, only much faster.

FL incorporates a simple, rule-based IF X AND Y THEN Z approach to a solving control problem rather than attempting to model a system mathematically. The FL model is empirically-based, relying on an operator's experience rather than their technical understanding of the system. For example, rather than dealing with temperature control in terms such as "SP =500F", "T <1000F", or "210C <TEMP <220C", terms like "IF (process is too cool) AND (process is getting colder) THEN (add heat to the process)" or "IF (process is too hot) AND (process is heating rapidly) THEN (cool the process quickly)" are used. These terms are imprecise and yet very descriptive of what must actually happen. Consider what you do in the shower if the

temperature is too cold: you will make the water comfortable very quickly with little trouble. FL is capable of mimicking this type of behavior but at very high rate. FL requires some numerical parameters in order to operate such as what is considered significant error and significant rate-of-change-of-error, but exact values of these numbers are usually not critical unless very responsive performance is required in which case empirical tuning would determine them. For example, a simple temperature control system could use a single temperature feedback sensor whose data is subtracted from the command signal to compute "error" and then time-differentiated to yield the error slope or rate-of-change-of-error, hereafter called "error-dot". Error might have units of degs F and a small error considered to be 2F while a large error is 5F. The "error-dot" might then have units of degs/min with a small error-dot being 5F/min and a large one being 15F/min. These values don't have to be symmetrical and can be "tweaked" once the system is operating in order to optimize performance. Generally, FL is so forgiving that the system will probably work the first time without any tweaking.

FL was conceived as a better method for sorting and handling data but has proven to be an excellent choice for many control system applications since it mimics human control logic. It can be built into anything from small, hand-held products to large computerized process control systems. It uses an imprecise but very descriptive language to deal with input data more like a human operator. It is very robust and forgiving of operator and data input and often works when first implemented with little or no tuning.

For the simulation of the real time working of the controller a **MAMDANI** model is used. A fuzzy logic controller is made using following specification-

- MATLAB version 7.0.0.19920(R14)
- Microsoft XP Service Pack 2

- 3 inputs (3 levels) and one output (5 level) of “trimf” type
- 27 rules with “and” connection

Fuzzy logic controller was designed and 27 rules were defined. Plots of membership function and surface plot of relationship between inputs and output is shown below.

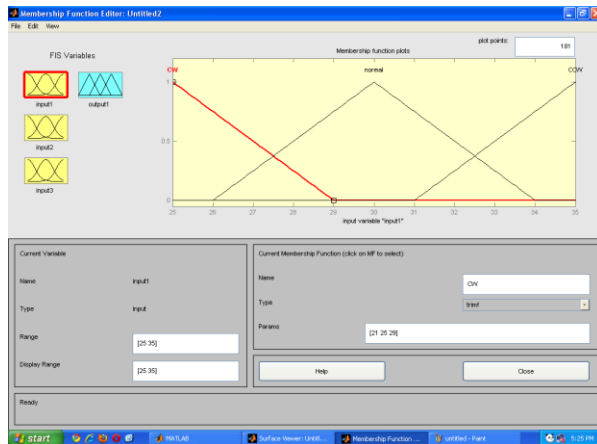


Fig. 1 Input 1

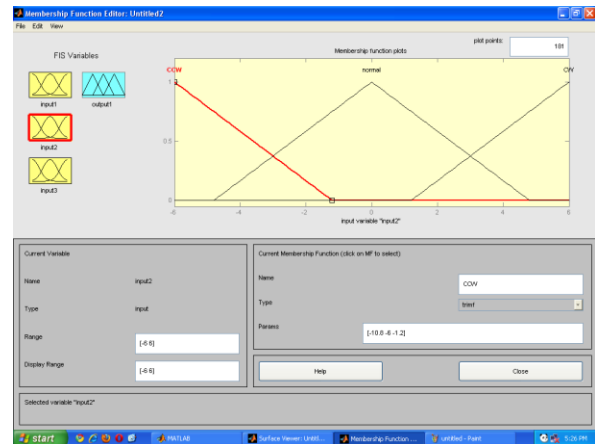


Fig. 2 Input 2

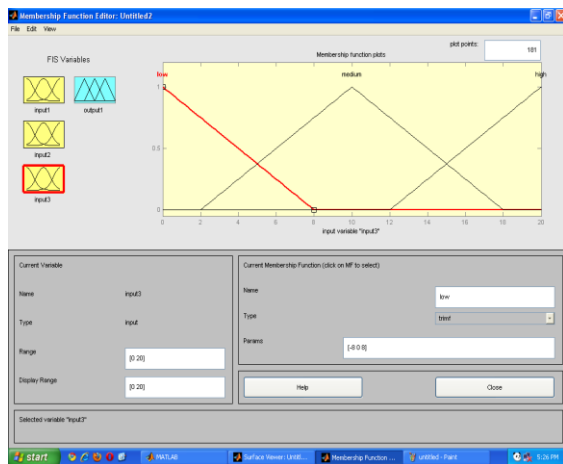


Fig. 3 Input 3

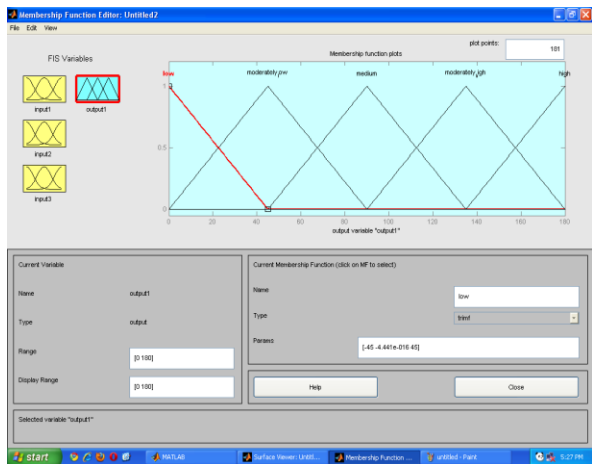


Fig. 4 Output 1

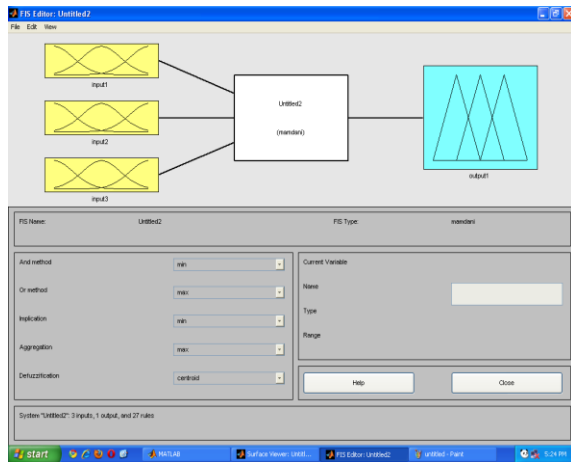


Fig. 5 FIS Properties

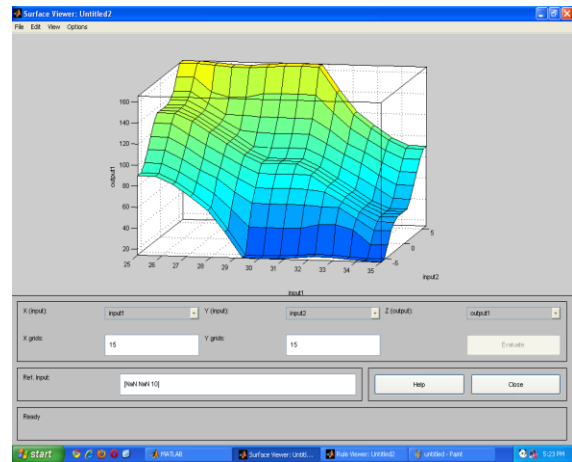


Fig. 6 Surface viewer

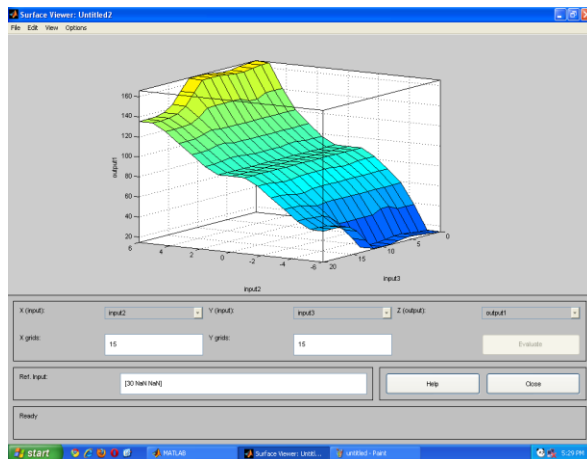


Fig. 7 Surface Viewer

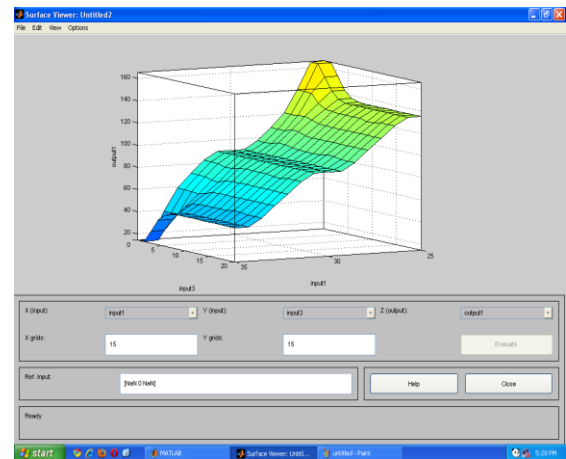


Fig. 8 Surface Viewer

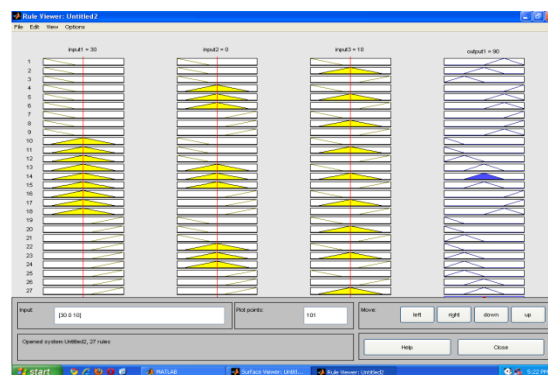


Fig. 9 Rules

ACTUATOR UNIT

As the robot tilts, we require applying a restoring force to return the robot vertical position. A reaction wheel pendulum model is followed for the balancing purpose. The components used are:

- High torque 12V DC motor
- A Control moment Gyroscope
- 1 μ f, 15V capacitor

Control Moment Gyroscope

A control momentum gyroscope (CMG) is an attitude control device generally used in spacecraft attitude control systems. A CMG consists of a spinning rotor and one or more motorized gimbals that tilt the rotor's angular momentum. As the rotor tilts, the changing angular momentum causes a gyroscopic torque that rotates the spacecraft.

CMGs differ from reaction wheels. The latter applies torque simply by changing rotor spin speed, but the former tilts the rotor's spin axis without necessarily changing its spin speed. CMGs are also far more power efficient. For a few hundred watts and about 100 kg of mass, large CMGs have produced thousands of newton meters of torque. A reaction wheel of similar capability would require megawatts of power.

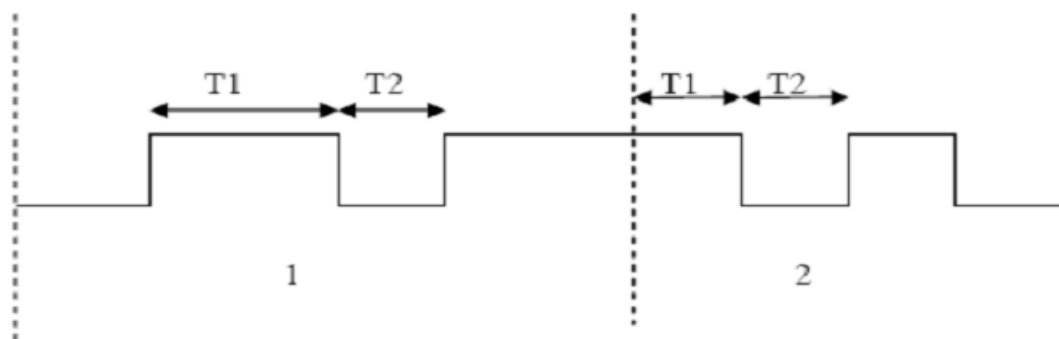
CMGs have been used for decades in large spacecraft, including the Skylab and Mir space stations and the International Space Station. The first generation of commercial imaging

satellites (QuickBird and Ikonos) used reaction wheels for attitude control. Today, only Ball Aerospace's WorldView-1 satellite built for DigitalGlobe (which provides data to Google Earth) is flying CMG's for attitude control. Ball Aerospace's next in the series, WV-2 will also use CMG's thus enabling the satellite to collect nearly 1 million sq kilometers per day.

The most effective CMGs include only a single gimbal. When the gimbal of such a CMG rotates, the change in direction of the rotor's angular momentum represents a torque that reacts onto the body to which the CMG is mounted, e.g. a spacecraft. Except for effects due to the motion of the spacecraft, this torque is due to a constraint, so it does no mechanical work (i.e. requires no energy). Single-Gimbal CMGs exchange angular momentum in a way that requires very little power, with the result that they can apply very large torques for minimal electrical input.

Motor Control

Pulse width modulation: PWM output is basically a series of pulses with varying size in pulse width. This PWM signal is output from the h-bridge circuit to control the wiper motor. The difference in pulse length shows the different output of h-bridge circuit controlling the output speed of the motor.



Pulse Width Modulation waveform

Figure above shows the varying pulse length of the pulse width modulation (PWM) scheme. Let's say that the PWM frequency is about 50 Hertz, with a period cycle of 20ms. Therefore assuming that the T1 and T2 length values are 15ms and 5ms respectively, the duty cycle can be calculated as below:

$$\text{Duty cycle} = T1 / (T1+T2) * 100\%$$

$$= 15/20 * 100\%$$

$$= 75\%$$

The capacitor connected across the motor charges and discharges during the on and off time respectively, thus behaving like an integrator. The torque generated by the motor is a function of the average value of current supplied to the motor.

CONCLUSION

During this project, there are a few experiments that are unaccomplished. Because of this, the concrete result is unable to attain and this is the main drawback that hampers the overall project. Therefore appropriate conclusions are not able to achieve. The problem with the oscillation still remains with the system and to achieve a stable solution, in future ore work has to be done. Complete design of the logic controller with the concept of both PID controller and the fuzzy logic certainly provided better results then using the either one separately.

FUTURE IMPROVEMENT

More accuracy can be achieved in the design if the mathematical modeling of the system is to be done using control system toolbox in Matlab. Defining more accurate rules for fuzzy logic controller and tuning of the PID controller can be done to improve the efficiency of the logical processing unit.

REFERENCES

- [1] Brennan, L. (1905) U.S. Patent No. 796, 893. Washington, D.C.: U.S. Patent and Trademark Office.
- [2] Carter, De Rubis, Guterrez, Schoellig, Stolar. "Gyroscopically Balanced Monorail System Final Report" (2005) Columbia University.
- [3] E. Ferreira, S. Tsai, C. Paredis, and H. Brown Advanced Robotics, Vol. 14, No. 6, June, 2000, pp. 459 - 475.
- [4] C.H. Ross, J. C. Hung, "Stabilization of an Unmanned Bicycle," Proc. IEEE Region III Convention, 1968, pp. 17.4.1-17.4.8.
- [5] Gallaspy, J. "Gyroscopic Stabilization of an Unmanned Bicycle." Ph.D. Thesis, Auburn University (2000).
- [6] Anderson, D.P, 'Nbot, a two wheel balancing robot', <[http://www.geology.smu.edu/~dpa
www/robo/nbot](http://www.geology.smu.edu/~dpa/www/robo/nbot)>
- [7] Steve Hassenplug, 2002, 'Steve's Legway', <<http://www.teamhassenplug.org/robots/legway/>>
- [8] Dean Kamen, 2001, <<http://www.segway.com>>
- [9] John Green, David Krakauer, March 2003, New iMEMS Angular Rate Sensing Gyroscope, <<http://www.analog.com/library/analogDialogue/archives/37-03/gyro.html>>
- [10] Peter Hemsley, 32-bit signed integer maths for PICS, <<http://www.piclist.com/techref/microchip/math/32bmath-ph.htm>>
- [11] Mosfets and Mosfet's drivers, <http://homepages.which.net/~paul.hills/SpeedControl/Mosfets.html>

- [12] Rick Bickle, 11 July 2003, 'DC motor control systems for robot applications',
<<http://www.dprg.org/tutorials/2003-10a/motorcontrol.pdf>>
- [13] Carnegie Mellon, 26 August 1997, 'Control Tutorials for Matlab', The University of Michigan,
<<http://www.engin.umich.edu/group/ctm/PID/PID.html>>
- [14] <<http://www.boondog.com/tutorials/mouse/mouseHack.htm>>
- [15] Martin Rowe, 11 January 2001, Measuring PWM motor efficiency, Test & Measurement World, <<http://www.reed-electronics.com/tmwworld/article/CA180848.html>>
- [16] Gerry, 6 February, Tilt sensors for your Robot,
<<http://www.roboticsindia.com/modules.php?name=News&file=article&sid=90>>
- [17] (c) 1998, 2001 EME Systems, Berkeley CA U.S.A., Pulse Width Modulation,
<http://www.emesystems.com/BS2PWM.htm>
- [18] Dennis Clark and Michael Owings, 'Building Robot Drive Trains', McGraw Hill Companies.
- [19] Naoji Shiroma, Osamu Matsumoto, Shuji Kajita, Kazuo Tani, 'Cooperative Behavior of a Wheeled Inverted Pendulum for Object Transportation', Proceedings of the 1996
- [20] IEEE/RSJ International conference on Intelligent Robots and Systems '96, IROS 96, volume: 2, 4-8 Nov. 1996 Pg(s): 396-401 vol.
- [21] Grasser, Felix, Alonso D'Arrigo, Silvio Colombi & Alfred C. Rufer, 2002, 'JOE: A Mobile, Inverted Pendulum', IEEE Transactions on Industrial Electronics, vol 49.
- [22] Young Soo Suh, 'Attitude Estimation using Low Cost Accelerometer and Gyroscope',
- [23] Proceedings of the 7th Korea-Russia International Symposium, KORUS 2003, Pg(s) 423-427.

[24] Albert-Jan Baerveldt and Robert Klang, 'A Low-cost and Low-weight Attitude Estimation System for an Autonomous Helicopter', Halmstad University, Sweden, Pg(s) 391-395.

[25] Yongjun Hou, Greg R.Luecke, October 5-8 2003, 'Control of the Tight Rope Balancing Robot', Proceedings of the 2003 IEEE International Symposium on Intelligent Control, Houston, Texas, Pg(s): 896-901.

[26] Alessio Salerno and Jorge Angeles, 'The Control of Semi-Autonomous Two-Wheeled Robots Undergoing Large Payload-Variations', Proceedings of the 2004 IEEE International Conference on Robotics & Automation, New Orleans, LA, Pg(s): 1740-1745.

[27] Kiyoshi Komoriya and Eimei Oyama, 'Position Estimation of a Mobile Robot Using Optical Fiber Gyroscope (OFG)', Pg(s): 143-149.

[28] Prof. John Billingsley, Mechatronics Practice Unit,
<<http://www.usq.edu.au/course/material/eng3905/>>

[29] Dr.Tony Ah Fock, Power Electronics study book 1 and 2, University of Southern Queensland.

[30] Paul E.Sandin, "Robot Mechanisms and Mechanical Devices Illustrated", The McGrawHill companies.

[31] J.J. D'Azzo, C.H. Houpis, Feedback Control System Analysis and Synthesis, second edition, McGrawHill International Editions. (ISBN 0-07-Y85150-6) Pg11

[32] 'Inverted Pendulum', Microrobot NA, <http://www.microrobotna.com/pendulum.htm>

[33] R.E Kalman, 1960, 'A New Approach to Linear Filtering and Prediction Problems', Transactions of the ASME- Journal of Basic Engineering, 82(Series D), pp35-45.

- [34] <<http://www.myengineeringsite.com/>>
- [35] <http://www.seattlerobotics.org/encoder/Mar98/fuz/fl_part1.html>
- [36] <<http://www.foretrade.com/Documents/FUZZY%20LOGIC.doc>>
- [37] <http://en.wikipedia.org/wiki/PID_controller>
- [38] <http://en.wikipedia.org/wiki/Kalman_filter>
- [39] <<http://www.srimca.edu.in/Srijan/Srijanoct2006/ITArticles/FuzzyLogic-Intro.htm>>
- [40] <http://en.wikipedia.org/wiki/Ultrasonic_transducer>
- [41] <http://en.wikipedia.org/wiki/Control_moment_gyroscope>
- [42] <<http://www.freepatentsonline.com/7621302.html>>
- [43] <<http://ethesis.nitrkl.ac.in/578/>>
- [44] <<http://www.faqs.org/patents/app/20090084194>>
- [45] <<http://www.freepatentsonline.com/7267085.html>>
- [46] <<http://www.faqs.org/patents/app/20090184744>>
- [47] <<http://www.faqs.org/patents/app/20090281431>>
- [48] <<http://www.faqs.org/patents/app/20090282916>>
- [49] <<http://www.freepatentsonline.com/y2010/0035518.html>>
- [50] <<http://www.faqs.org/patents/app/20090043550>>

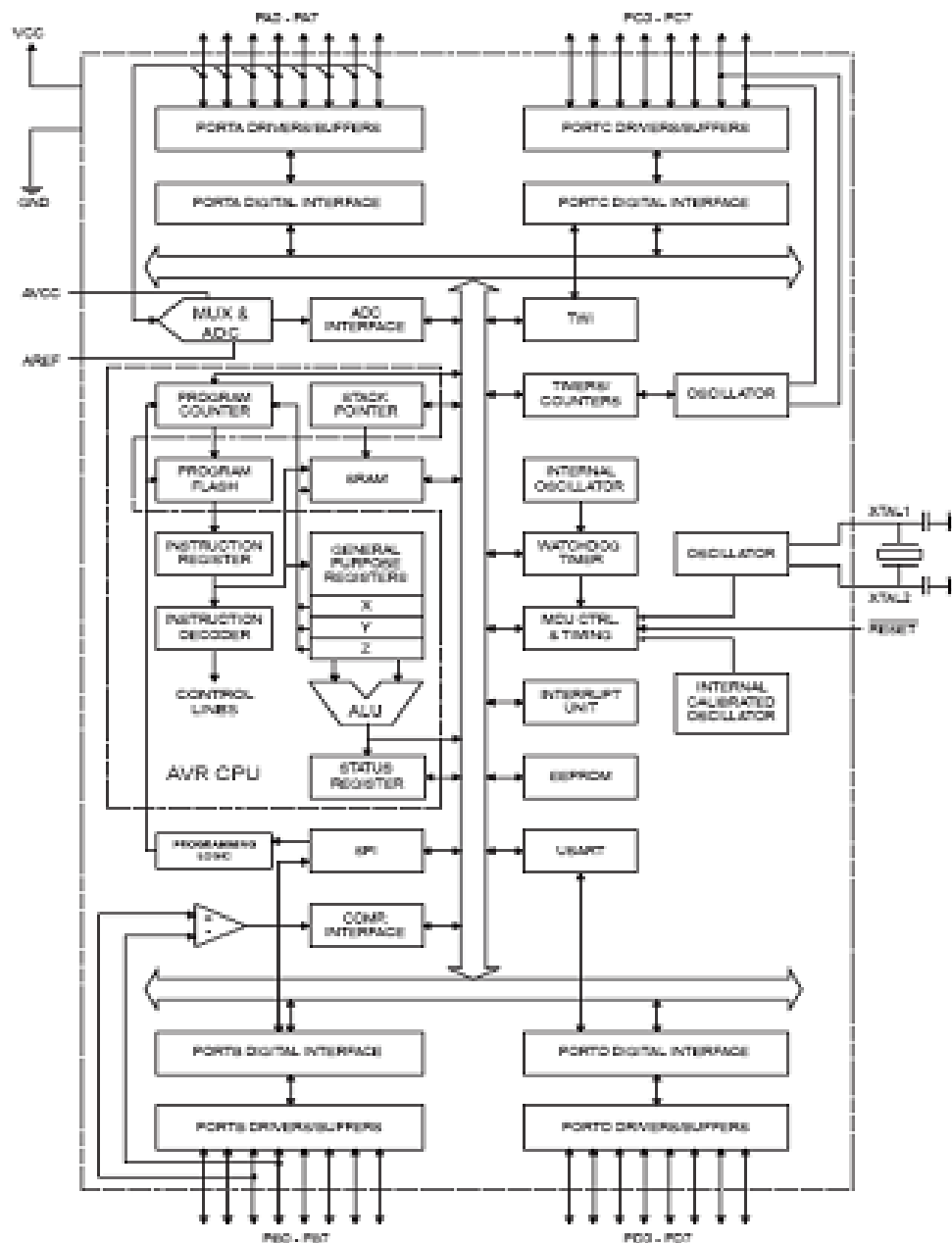
APPENDIX-1

Features of ATmega-16

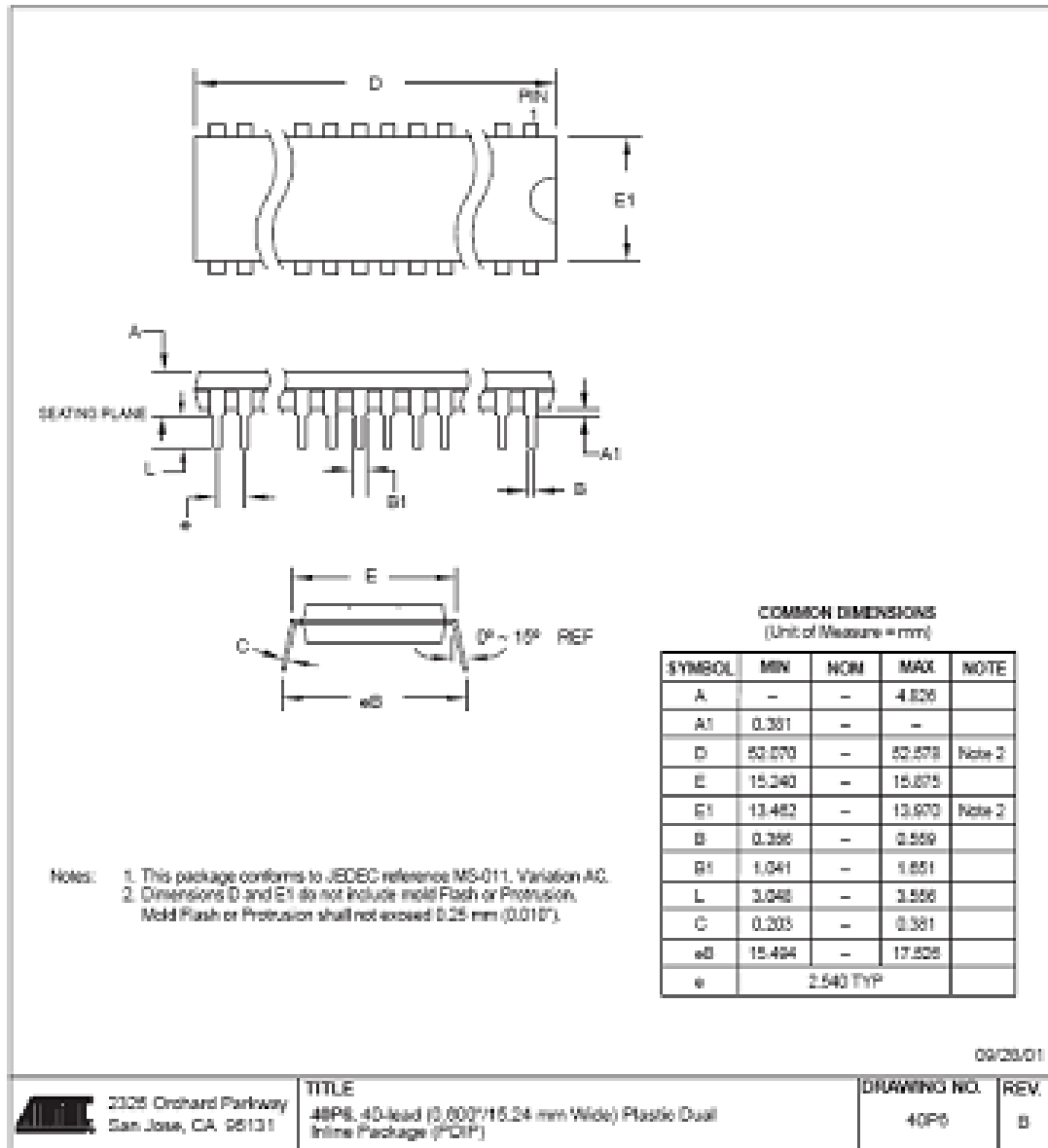
- High-performance, Low-power AVR® 8-bit Microcontroller
- Advanced RISC Architecture
 - 131 Powerful Instructions – Most Single-clock Cycle Execution
 - 32 x 8 General Purpose Working Registers
 - Fully Static Operation
 - Up to 16 MIPS Throughput at 16 MHz
 - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory segments
 - 16K Bytes of In-System Self-programmable Flash program memory
 - 512 Bytes EEPROM
 - 1K Byte Internal SRAM
 - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
 - Data retention: 20 years at 85°C/100 years at 25°C(1)
 - Optional Boot Code Section with Independent Lock Bits
- In-System Programming by On-chip Boot Program True Read-While-Write Operation
 - Programming Lock for Software Security
- JTAG (IEEE std. 1149.1 Compliant) Interface
 - Boundary-scan Capabilities According to the JTAG Standard
 - Extensive On-chip Debug Support
 - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
 - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
 - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
 - Real Time Counter with Separate Oscillator
 - Four PWM Channels
 - 8-channel, 10-bit ADC
 - 8 Single-ended Channels
 - 7 Differential Channels in TQFP Package Only
 - 2 Differential Channels with Programmable Gain at 1x, 10x, or 200x
 - Byte-oriented Two-wire Serial Interface
 - Programmable Serial USART
 - Master/Slave SPI Serial Interface
 - Programmable Watchdog Timer with Separate On-chip Oscillator
 - On-chip Analog Comparator
- Special Microcontroller Features
 - Power-on Reset and Programmable Brown-out Detection
 - Internal Calibrated RC Oscillator
 - External and Internal Interrupt Sources
 - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
- I/O and Packages
 - 32 Programmable I/O Lines
 - 40-pin PDIP, 44-lead TQFP, and 44-pad QFN/MLF
- Operating Voltages
 - 2.7 - 5.5V for ATmega16L
 - 4.5 - 5.5V for ATmega16
- Speed Grades
 - 0 - 8 MHz for ATmega16L
 - 0 - 16 MHz for ATmega16
- Power Consumption @ 1 MHz, 3V, and 25°C for ATmega16L
 - Active: 1.1 mA
 - Idle Mode: 0.35 mA
 - Power-down Mode: < 1 µA

Internal block diagram of ATmega16

Figure 2. Block Diagram



Packaging Information



APPENDIX-2

Accelerometer module specification

Chip-freescale semiconductors

Board-Robokits

Board size - 28mm X 23mm

5 pin interface (VCC, GND, Xout, Yout, Zout)

Selectable Sensitivity (1.5g/2g/4g/6g) and Sleep Mode Selectable through jumpers or microcontroller

Current Consumption: 500 μ A

Low Voltage Operation: 2.6V to 5V

High Sensitivity (800 mV/g @ 1.5g) for small movements

Fast Turn On Time

Integral Signal Conditioning with Low Pass Filter

Robust Design, High Shocks Survivability

Board Top Layout

